Available online at www.sciencedirect.com
SCIENCE DIRECT®
Theoretical Computer Science 322 (2004) 335–354
Theoretical Computer Science
www.elsevier.com/locate/tcs

Real-time recognition of languages on an two-dimensional Archimedean thread

M. Delorme, J. Mazoyer*
LIP, Ecole Normale Supérieure de Lyon, 46 Allée d'Italie, Cedex 07, F-69364 Lyon, France

Abstract

Fine studies on how powerful are two-dimensional computing devices are restricted by the difficulty to set up a relevant notion of language of figures. In the case of two-dimensional cellular automata, several solutions have been proposed, but we adopt here another point of view: to the automaton is added a one-to-one "locally connected" mapping, τ, from $\mathbb{Z}$ (or $\mathbb{N}$) onto $\mathbb{Z}^2$, we have named a thread. That leads to a notion of language (usual language of words) recognition depending on a thread, what in turn leads to complexity classes, especially classes of languages recognized in real time on such devices. Here we consider two-dimensional cellular automata with the discrete Archimedean spiral as thread, and we compare the class of languages recognized in real time by two-dimensional cellular automata with the Archimedean spiral to the class of languages recognized in real time by one-dimensional classical cellular automata.

Keywords: Cellular automata; Language recognition; Complexity; Real time

1. Introduction

Cellular automata constitute well defined parallel devices. Simple local rules often generate very complex global behaviors. Understanding this complexity is important, what is mainly pursued, at least for cellular automata, from two points of view. First, cellular automata are considered as discrete dynamical systems, then, for example, starting from significant sets of initial configurations, one looks to predict some characteristics of their evolutions. Secondly, they are viewed as a massively parallel computation model, then, for example, one studies related relevant complexity classes and tries to

* Corresponding author.
E-mail addresses: marianne.delorme@ens-lyon.fr (M. Delorme), jacques.mazoyer@ens-lyon.fr (J. Mazoyer).

0304-3975/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2004.03.016

compare them with complexity classes being due to other computation models. That is this second point of view that we adopt in the present paper.

Complexity classes are classes of languages which encode problems. Unfortunately, if language recognition is natural for one-dimensional cellular automata (as well as for other computational models on which the order of the input letters is necessarily respected thanks to the structure of the devices space), it is no more the case for two-dimensional cellular automata since the way to place the letters on the plane is not necessary, at least when one demands that the whole word appear on the plane at initial configuration, that is for parallel mode recognition. Actually, there is another mode of recognition for cellular automata: the sequential mode for which inputs are given letter by letter on a specified cell. If sequential mode gives birth to classes the relations of which are well known [2], parallel mode is more difficult to study, leads to classes whose some relations are still open, as, for example, to know whether the class of languages recognized in linear time on one-dimensional cellular automata is equal to the one of languages recognized in real time. [1] A landscape of the complexity classes in dimension one is described, for example, in [3].

Let us come back to the two-dimensional case for parallel mode. Mainly three ways of arranging words on the plane have been proposed: to linearly put the letters into the $(0,0),\ldots,(n-1,0)$ cells [5], to cut the words into factors of equal length and arrange these factors inside a rectangle, letters in increasing order from left to right for each line as in [5] or one time out of two as in [8]. In [4], we have introduced a notion of "thread" along which the words are set in their natural order, the whole composing the significant part of the initial configuration of some two-dimensional cellular automaton. With a convenient notion of recognition, we get classes of languages which can be compared to known ones on other devices (one-dimensional cellular automata or Turing machines). Clearly, the recognition depends on both the automaton and the thread, so we are also really studying interactions between two-dimensional cellular automata and threads via language recognition, especially real-time recognition.

The important point is the notion of real time. As in many papers, see for example [1,7], with real time one understands "as soon as possible", that means the cell which decides to accept or reject an input word has knowledge of the whole word. In dimension one, real time is then n or $n+1$ according to the fact that the last cell knows that it is the last one or not. In dimension two, the real time being when the accepting cell receives information that no new letter of the input word will be received, real time may be less than the length of the word. In our case, real time depends both on the thread and the considered neighborhood.

In the following, we choose a very simple thread, that we name the Archimedean thread, and the Moore neighborhood. [2] We proved in [4] that any rational (regular) language is recognized in real time with the Archimedean thread. Here, we prove that real-time recognition on two-dimensional cellular automata with the Archimedean

[1] Let us remind that one works in space bounded by the length of the word.

[2] It is easier, here, to work with the Moore's neighborhood. Besides, if a language is recognized on some cellular automaton with the Von Neumann neighborhood, it is recognized by another cellular automaton with the Moore's one.

thread is strictly less powerful than real-time recognition on one-dimensional cellular automata.

We will first give useful definitions, then show that if $\mathscr{R}_\chi$ denotes the class of languages recognized in real time on two-dimensional cellular automata with the Archimedean thread, if $\mathscr{R}_{1D}$ denotes the class of languages recognized in real time on one-dimensional cellular automata, $\mathscr{R}_\chi \neq \mathscr{R}_{1D}$. Finally, we will explain $\mathscr{R}_\chi \subset \mathscr{R}_{1D}$, what is involved enough.

2. Definitions

Let us first specify some requisite definitions,[3] and note, once and for all, that the neighborhood for all cellular automata is the Moore's one.

The Archimedean thread is obtained from a curve $\mathscr{C}$, which is described below. For all $(x, y) \in \mathbb{Z}^2$, let $C_{(x,y)}$ denote the unit square the south-west corner of which is (x, y).

Definition 1. (1) Starting from the center of $C_{(0,0)}$, one draws an horizontal line up to the center of $C_{(1,0)}$, a vertical one up to the center of $C_{(1,1)}$, a horizontal one up to the center of $C_{(-1,1)}$, a vertical one up to the center of $C_{(-1,-1)}$ and finally a horizontal one up to the center of $C_{(1,-1)}$.

The process is then iterated: starting from the point (h,h), one draws the straight lines to the successive points $(h+1,h),(h+1,h+1),(-(h+1),h+1),(-(h+1),-(h+1))$ and finally $(h+1,-(h+1))$.

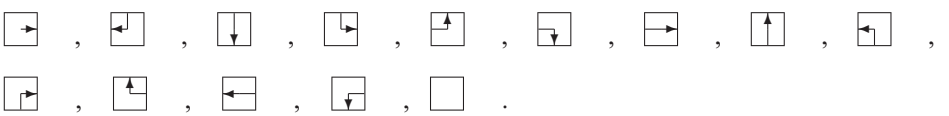
(2) The Archimedean thread is the one-to-one mapping χ from $\mathbb{N}$ onto $\mathbb{Z}^2$, shown in Fig. 1(a), and defined by

- $\chi(0) = (0,0)$,
- if, for all j, $j \in \{0,\ldots,i\}$, $\chi(j)$ is defined, $\chi(i+1)$ is the south-west corner of the unique square
  - whose south-west corner does not belong to $\chi(\{0,\ldots,i\})$,
  - and which is connected by $\mathscr{C}$ to the unique square $C_{\chi(i)}$.

The converse χ^{-1} of χ gives a numbering of the cells along the thread represented in Fig. 1(b).

Definition 2. A two-dimensional cellular automaton $\mathscr{A}$ is an ordered pair (S,δ) where S is the finite set of states and δ, from S^9 into S, the local transition function. Then,

- A two-dimensional cellular automaton with the Archimedean thread $\mathscr{A}_\chi$ is some $(S^\star,\delta)$ with $S^\star = S_\chi \times S$ where S_χ is made of the 14 following states which represent the Archimedean thread:



[3] More on threads may be found in [4].

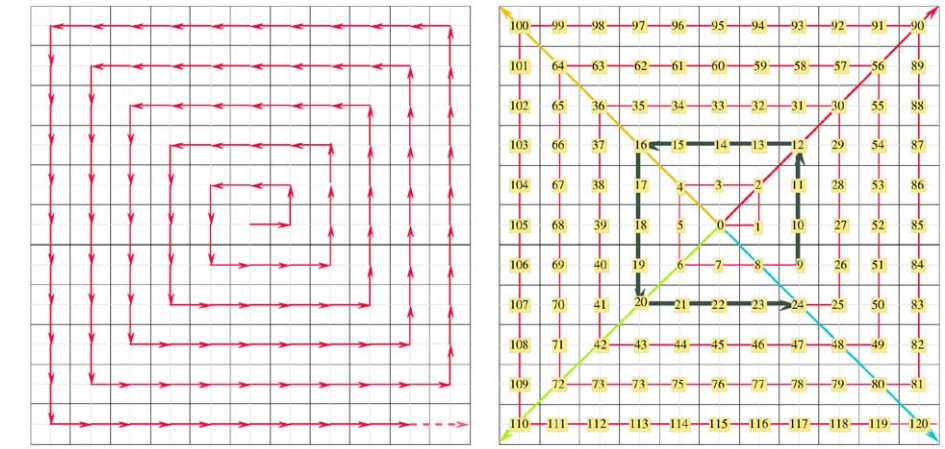


Fig. 1. The Archimedean thread: (a) the Archimedean curve, (b) numbering of the cells along of the Archimedean thread: χ^{-1}.

An arrow starting from $\chi(i)$ comes from $\chi(i-1)$ and indicates the direction of $\chi(i+1)$. The state transition function is from $S^{\star^9}$ into $S^\star$ and never changes the component of S_χ. States of S_χ are called thread-components and are invariant.

- A configuration of $\mathscr{A}_\chi$ is an application from $\mathbb{Z}^2$ into $S^\star$ such that the component in S_χ represents the Archimedean thread. From a configuration C at time t, the cellular automaton enters at time $t+1$ the configuration $C^\star$ defined by: $i,j \in \mathbb{Z}$ and $C^\star(i,j) = \delta(C(i,j), C(i+1,j), C(i+1,j+1), C(i,j+1), C(i-1,j+1), C(i-1,j), C(i-1,j-1), C(i,j-1), C(i+1,j-1))$.
  The function which associates to the configuration C the configuration $C^\star$ previously defined is the global function of the cellular automaton.
  The evolution of the cellular automaton from an initial configuration, configuration of the network at time $t=0$, is the sequence $(C_t)_{t\in\mathbb{N}}$ of its successive configurations. C_t is called the configuration at time t and state $C_t(i,j)$ of the cell (i,j) at time t will be denoted by $\langle i,j\rangle_t$.

Definition 3. Let L be a language on an alphabet Σ and let $\mathscr{A}_\chi = (S^\star, \delta)$ a two-dimensional cellular automaton with the Archimedean thread. We say that L is recognized by $\mathscr{A}_\chi$ when:

- $\Sigma \subseteq S$,
- S contains two distinguished disjoint subsets S_{accept} and S_{reject} as well as a quiescent state s_q, which then satisfies $\delta((\chi_c, s_q), \ldots, (\chi_{se}, s_q)) = (\chi_c, s_q)$, where χ_c denotes the thread component of the state of cell c, χ_i and χ_{ij} the thread components of the states of the c-neighbors $(i,j \in \{e,n,w,s\})$,

- For any word $a_0 \dots a_{\ell-1}$ on Σ, starting from the initial configuration defined by $\langle \chi_{\chi(i)}, s_{\chi(i)} \rangle_0 = (\chi_{\chi(i)}, a_i)$ for $i \in \{0, \dots, \ell - 1\}$, with

  $\chi_{\chi(0)} =$  , $\chi_{\chi(i)} \in S_\chi - \{$  $\}$

  for $i \in \{1, \dots, \ell - 1\}$, the other cells being in state

  $\langle$  $, s_q \rangle_0$,

  there exists some time θ such that
  - $\langle 0, 0 \rangle_\theta = (x, s_{\chi(0)})$ where $x \in S_\chi$, $s_{\chi(0)} \in S_{\text{accept}}$ if and only if the word $a_0 \dots a_{\ell-1}$ belongs to L, else, $s_{\chi(0)} \in S_{\text{reject}}$,
  - and for any time t, $t < \theta$, $s_{\chi(0)}$ does not belong to S_{accept} nor to S_{reject}.

Finally, we have to specify real-time recognition in that framework. The distance to the origin $(0,0)$ of a point (x, y) is $\text{Max}(|x|, |y|)$ and the real time is the minimum time in which the cell at the origin knows that the input word has been in full read.

Definition 4. A language L is recognized in real time by a two-dimensional cellular automaton with the Archimedean thread when the recognition time for a word $a_0 \dots a_{\ell-1}$ is $\max_{i \in \{0,\dots,\ell-1\}} \{\chi_x(i), \chi_y(i)\} + 1$, where $\chi(i) = (\chi_x(i), \chi_i(h))$.

Let us remind that $\mathscr{R}_\chi$ (resp. $\mathscr{R}_{1D}$) denotes the class of languages recognized in real time by a two-dimensional cellular automaton with the Archimedean thread (resp. by a one-dimensional cellular automaton). We will now prove that they are different.

3. $\mathscr{R}_\mathscr{A} \neq \mathscr{R}_{1D}$

Let us first put to light some points of the Archimedean thread which plays a very special role as markers in the following: the "corners" of the spires. Let k, $k \geqslant 1$, be the number of a spire, then $(k, -k)$—the south-east corner (and the end) of the spire—is numbered $4k(k+1)$, (k, k)—the north-east corner of the spire—is numbered $4k(k-1) + 2k$, $(-k, k)$—the north-west corner of the spire—is numbered $4k^2$ and $(-k, -k)$—the south-west corner of the spire—is numbered $4k(k-1) + 6k$.

Proposition 5. *Real time to recognize a word of length n on a two-dimensional cellular automaton with the Archimedean thread is*

$$\mathscr{R}_\chi(n) = 1 + \max_{d \in \mathbb{N}} \{d/(2d+1)^2 \leqslant n\}.$$

Proof. As the distance to the origin of (x, y) is $\max\{|x|, |y|\}$, bawls $\mathscr{B}_{(0,0)}(d)$ are squares centered in $(0,0)$, with length side $2d + 1$. Let n be an integer such that $\chi(n) \in \mathscr{B}_{(0,0)}(d)$ and $\chi(n) \notin \mathscr{B}_{(0,0)}(d-1)$, $d \geqslant 1$. Time needed for information on cell numbered $\chi(n)$ to reach cell $(0,0)$ is d. Time needed by cell $(0,0)$ to know it has

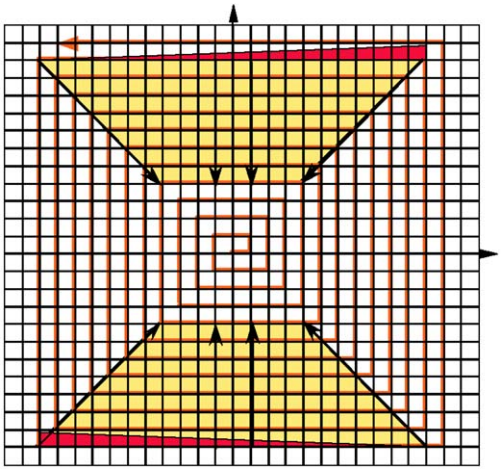


Fig. 2. The language L.

received all information on the input word is $d+1$ because, at that time, data arriving will contain the state

$$(\square, s_q).$$

The number of points inside $\mathscr{B}_{(0,0)}(d)$ is $(2d+1)^2$, so the first point of the Archimedean thread outside $\mathscr{B}_{(0,0)}(d)$ is its $(1+(2d+1)^2)$th point, image of $(2d+1)^2$ by χ. □

Proposition 6. *Let L be the language on $\{0,1\}$ made of words*

$$0^{2n(2n-1)}a_1 \dots a_{2n+1}0^{2n-1}a_1 \dots a_{2n+1}$$

for $n \geqslant 1$. L is a language which belongs to $\mathscr{R}_{1D}$ but not to $\mathscr{R}_\chi$.

Proof. We first observe that each word $0^{2n(n-1)}a_1 \dots a_{2n+1}0^{2n-1}a_1 \dots a_{2n+1}$ is of length $(2n+1)^2$, which is the number of cells along the thread from the origin to the south-east corner of the nth spire (point $(n,-n)$). Moreover, the letters $a_1,\dots,a_{2n+1}$ of the first factor are on the cells numbered from $4n(n-1)+2n$ to $4n^2$, those of the second factor are on the cells numbered from $4n(n-1)+6n$ to $4n(n+1)$ (see Fig. 2).

Let us suppose that L is real-time recognized by a cellular automaton with the Archimedean thread, $\mathscr{A}_\chi=(Q,\delta)$. After m steps, the initial configuration determined by word $0^{2n(2n-1)}a_1 \dots a_{2n+1}0^{2n-1}b_1 \dots b_{2n+1}$ ends up in a configuration the "useful" part of which, $C_{(a,b,m)}$, is made of points (x,y) such that $|x| \leqslant n-m+1$ and $|y| \leqslant n-m+1$. Let us denote $C^{\geqslant}_{(a,m)}$ (resp. $C^{<}_{(b,m)}$) the points of $C_{(a,b,m)}$ with non-negative (resp. negative) ordinates. States of $C^{\geqslant}_{(a,m)}$ only depend on $a_1 \dots a_{2n+1}$ and states of $C^{<}_{(b,m)}$ only depend on $b_1 \dots b_{2n+1}$. Now, let us fix an integer k. There are $k(2k+1)$ points in $C^{<}_{(b,n-k)}$, and thus $|Q|^{k(2k+1)}$ possible distinct configurations $C^{<}_{(b,n-k)}$. But there are 2^{2n+1} words $b_1 \dots b_{2n+1}$, so, for n large enough, there exist two distinct words $b^\star_1 \dots b^\star_{2n+1}$

and $b_1^{\#} \dots b_{2n+1}^{\#}$ which lead to the same configuration $C^{<}_{(b,n-k)}$. Then the two words $0^{2n(2n-1)} b_1^{\star} \dots b_{2n+1}^{\star} 0^{2n-1} b_1^{\star} \dots b_{2n+1}^{\star}$ and $0^{2n(2n-1)} b_1^{\star} \dots b_{2n+1}^{\star} 0^{2n-1} b_1^{\#} \dots b_{2n+1}^{\#}$ lead to the same configurations $C^{\geqslant}_{(b^{\star},n-k)}$ and $C^{<}_{(b^{\star},n-k)}$ and consequently, after k steps, to the same state on the cell $(0,0)$ which must be both an accepting and a rejecting state, what is a contradiction.

It remains to show that L is recognized in real time by some one-dimensional cellular automaton. The method, illustrated in Fig. 3, is the one developed in [6]. The

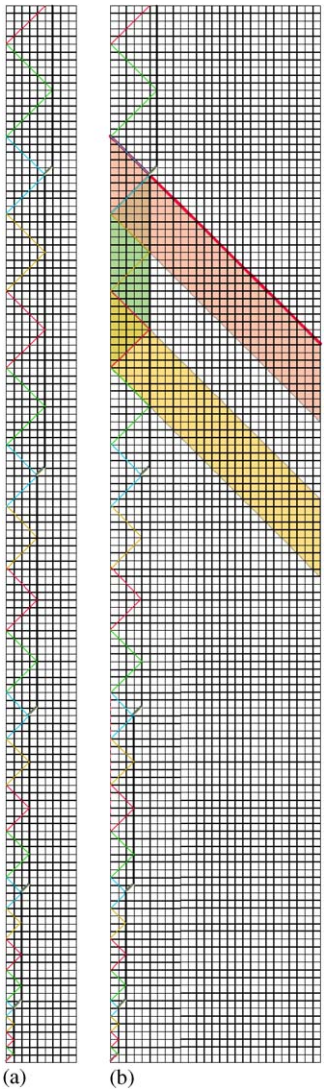


Fig. 3. Recognition of language L in proof of Proposition 6 by a one-dimensional cellular automaton: (a) a signal in $\sqrt{n}$, (b) recognizing language L.

basic point is the ability for a cellular automaton to mark, on the origin cell, times $4k(k+1)$, $4k(k-1)+2k$, $4k^2$, and $4k(k-1)+6k$ (which correspond to the numbers of the corners of the spires of the Archimedean thread). That allows to master the structure of the words in L which is actually determined by the structure of the spires of the Archimedean thread. The way to obtain the wanted marks on the origin cell is simple as it is shown in Fig. 3(a). A signal Σ is built in the following way: for $k \geqslant 1$, it stays $8k$ times on the cell k, what is measured by four successive signals which oscillate between cells 0 and k at maximum speed marking cell 0 each $2k$ times. So, the origin cell can keep in its state one of the four parts of a virtual spire. Afterwards, signal Σ moves to cell $k+1$ where the process is iterated.

Moreover, letters[4] of the input words (of length $(2n+1)^2$) run at maximal speed to the origin. When one of them, s, reaches it, if the origin is in a state which corresponds to the second quarter of a spire, what means that s represents a letter a_i of the possible first factor $a_1 \ldots a_{2n+1}$, a new signal is sent at maximal speed which bounces one time on Σ, comes back to the origin, generates there another one which runs at maximal speed and meets on Σ the signal coming from the letter at the same position i in the possible second factor $a_1 \ldots a_{2n+1}$ of the input word. If they represent different letters, a signal is sent to the origin cell which enters a rejecting state. Else, a signal is sent to the origin which says that the process is going on. Then the origin enters an accepting state when the end of the word arrives at the end of the last quarter of a virtual spire, see Fig. 3(b). □

Now, in order to get our result, we have to prove the classes inclusion. It is the subject matter of the next section.

4. $\mathscr{R}_{\mathscr{A}} \subset \mathscr{R}_{1D}$

Theorem 7. $\mathscr{R}_{\mathscr{A}} \subsetneq \mathscr{R}_{1D}$.

Proof. Let us then suppose that L is recognized in real time by some two-dimensional cellular automaton with the Archimedean thread $\mathscr{A}_\chi$. Using this knowledge, we have to build a one-dimensional automaton $\mathscr{B}$ which recognizes L in real time. To elaborate on the solution is long and technical enough. Thus, we start with a general view of the algorithm.

(a) *Overall process*: Starting from the initial configuration determined by some word w of length n, placed letter by letter on cells numbered 0 to $n-1$, $\mathscr{B}$ has to simulate the behavior of $\mathscr{A}_\chi$ on the same word. That will be done in generating a family of signals (actually thick signals as it will appear later) $(B_t)_{t \geqslant 0}$, each B_t representing the tth computation step of $\mathscr{A}_\chi$ on w, that means information on the letters and on the structure of the thread (see Fig. 4(a) and (b)).

B_0 is defined starting from a signal S_2 of slope 2 sent from the origin. On $\mathscr{B}$ the letters of the word are sent, at maximal speed to the origin. They meet S_2 which will

[4] In fact, signals representing letters.

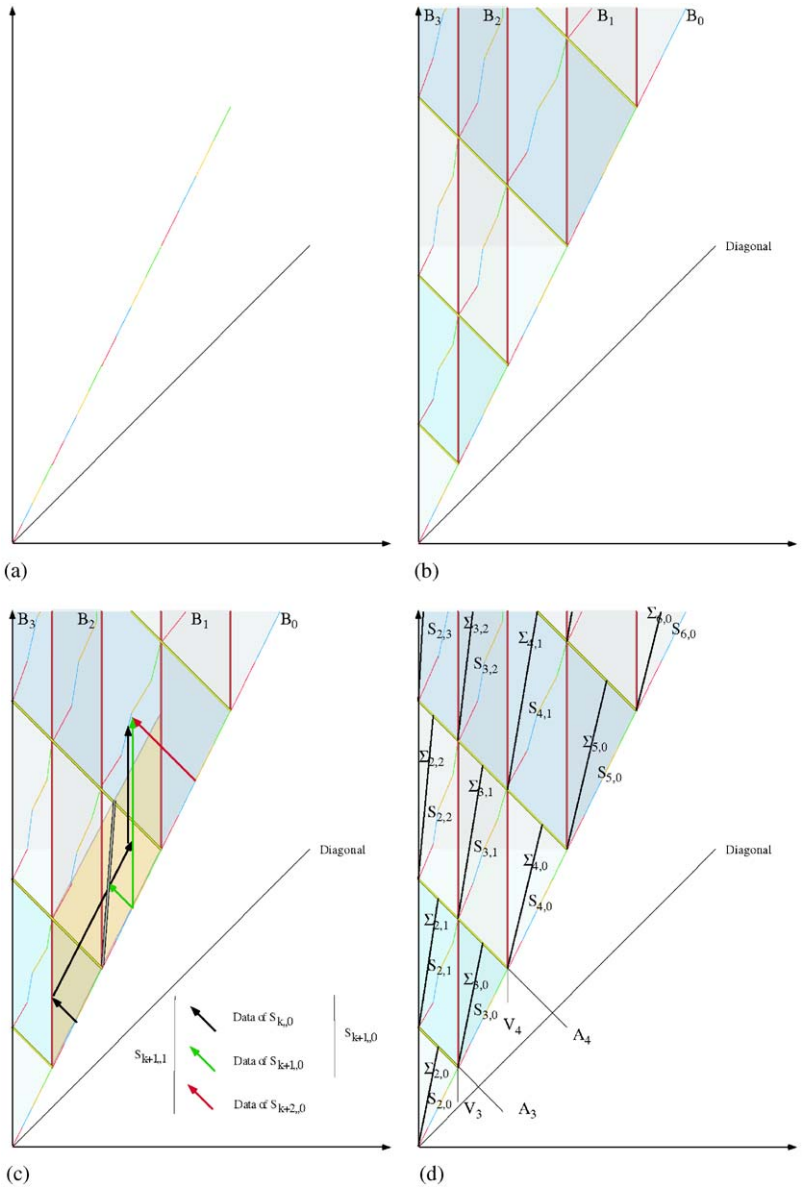


Fig. 4. How to prove Theorem 7: (a) signal carrying structure of the Archimedean thread; (b) moves of information; (c) flows of data entering the computations of sets $S_{n,t}$; (d) setting up of $\Sigma_{n,t}$.

capture, as its slope is 2, on one site, information arriving from three initial cells. So, in fact, $\mathscr{B}$ will send, from B_0, groups of three initial data. But B_0 has also to reproduce the structure of the thread, that is to mark the spires and also the four parts of each

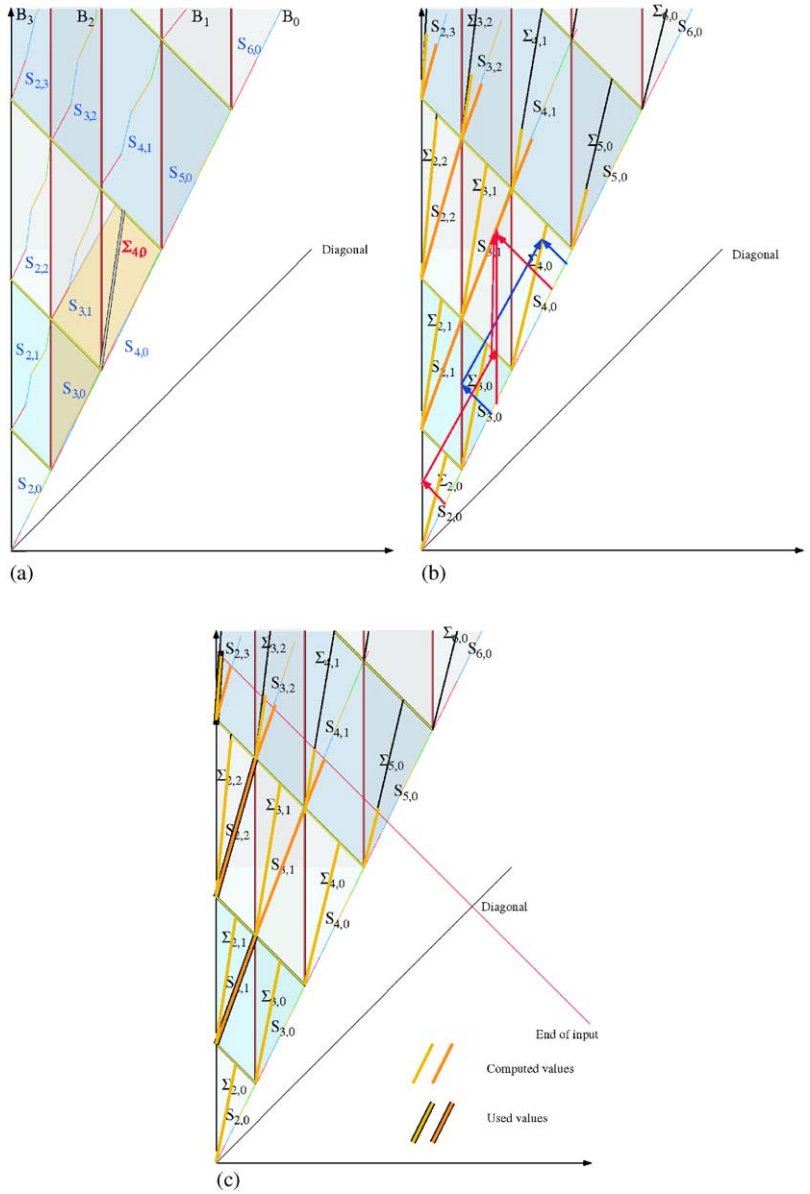


Fig. 5. How to prove Theorem 7 (continued): (a) locations of values of $\Sigma_{n,t}$; (b) needed values to get $\Sigma_{n,t}$; (c) the end of word.

spire, in such a way that one knows where are situated the "corners" of the spire as well as its first and last cell—points $(k, -(k+1))$ and $(k, -k)$ if k is the number of the spire.

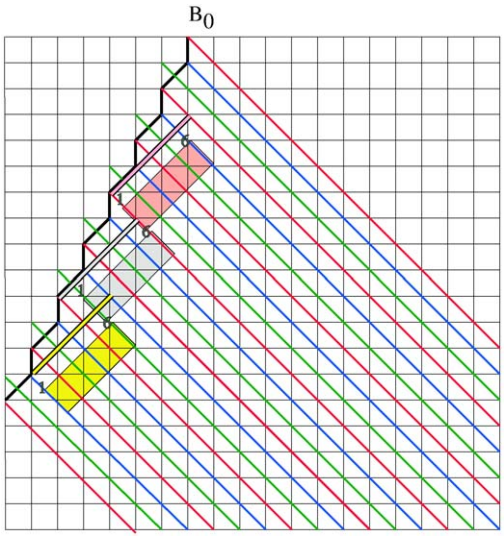


Fig. 6. Setting up signal B_0 (the three types of initialization).

When a cell knows that it is the "beginning" of some spire k, $k \geqslant 2$, it sends a vertical signal V_k and a second one, A_k, at maximal speed to the origin cell. On the space-time diagram of $\mathscr{B}$, V_k and V_{k+1} on the one hand, A_{k+1} and A_{k+2} on the other hand allow to determine the place corresponding to the spire k on B_t. Information carried on B_t in the space corresponding to spire k will be given by means of $\mathscr{B}$-states the set of which is denoted $S_{k,t}$. Computations of the values of states in $S_{k,t}$ are achieved by means of data flows as indicated in Fig. 4(c).

Up to now, we have not taken into account our aim to get the recognition in real time. On $\mathscr{A}_\chi$, the end of the input word w appears on a spire, let say v, and real time is obtained thanks to data coming from the $(v+1)$th spire. If, in the proposed simulation we have to wait for information from the $(v+1)$th spire, we will not get real time. Consequently, we have to set up a mechanism which will anticipate the fact that no input letter will appear on the next spire or, say otherwise, that the current spire is the one on which occurs the end of the input word. So, from each cell marking the beginning of a spire k on B_t starts a signal $\Sigma_{k,t}$ meaning that no input letter lays on the $(k+t+1)$th spire at initial time (Figs. 4(d) and 5(a)). States of $\Sigma_{k,t}$ are computed at the first meeting of the data flow coming from $S_{k-1,t-1}$ and $S_{k,t-1}$ as indicated in Fig. 5(b). Finally, Fig. 5(c) illustrates how the origin cell is able to know in real time data necessary to decide whether the initial word belongs or not to L.[5]

In the sequel, we are explaining in more details some important points of the simulation.

[5] Do notice that Figs. 4 and 5 already take into account that the first spire is specially dealt with (see point (f)).

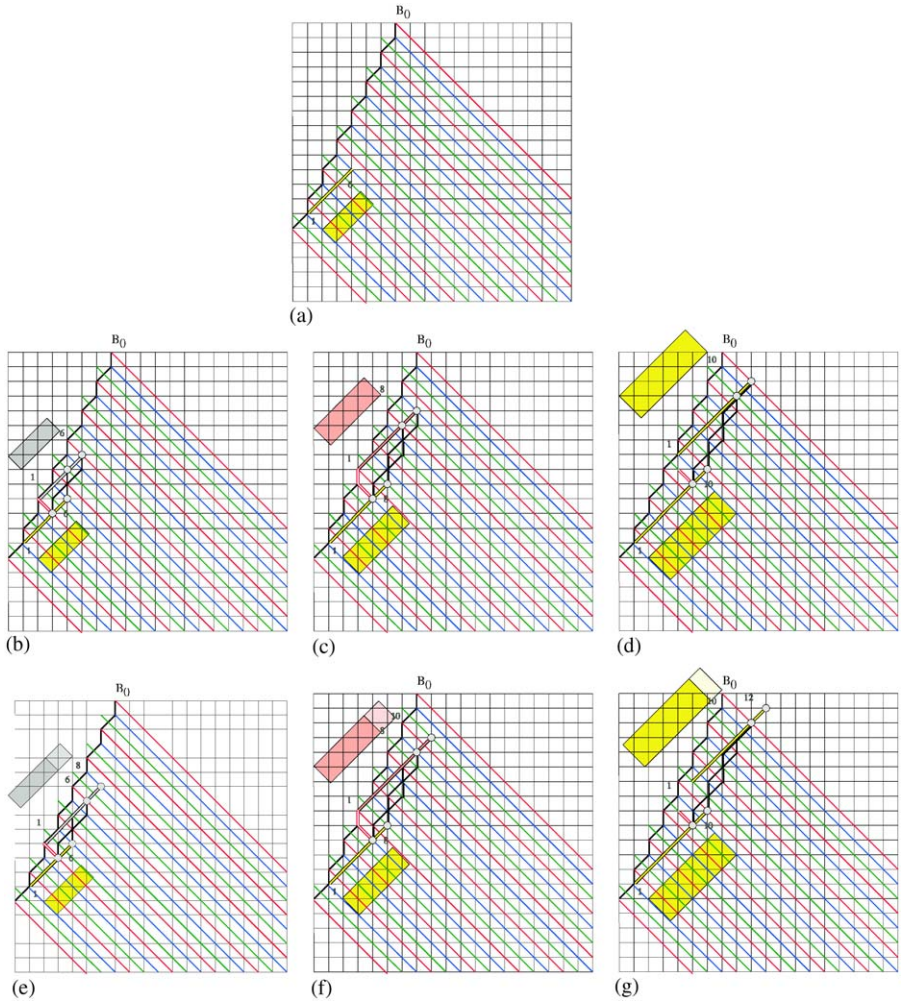


Fig. 7. Setting up signal B_0 (signal of type 1): (a) signal of type 1 of length $2k$, $T_{1,k}$; (b) signal $T_{1,k}$ giving another signal of length $2k$, $T_{2,k}$; (c) signal $T_{1,k}$ giving another signal, $T_{3,k}$, of length $2k$; (d) signal $T_{1,k}$ giving another signal, $T_{1,k}$, of length $2k$; (e) signal $T_{1,k}$ giving another signal, $T_{2,k+1}$, of length $2(k+1)$; (f) signal $T_{1,k}$ giving another signal of length, $T_{3,k+1}$, $2(k+1)$; (g) signal $T_{1,k}$ giving another signal, $T_{1,k+1}$, of length $2(k+1)$.

(b) *Signal* B_0: It is obtained in marking on S_2 the successive spires and their structure. That means to distinguish the beginning of each spire of number k, $k \geqslant 2$, and its four "sides" of length $2k$. But that also means to find an algorithm which allows, starting from one segment of length $2k$ to build the three next one in such a way that the last one of them were able to generate the first one of length $2(k+1)$ of the next spire.

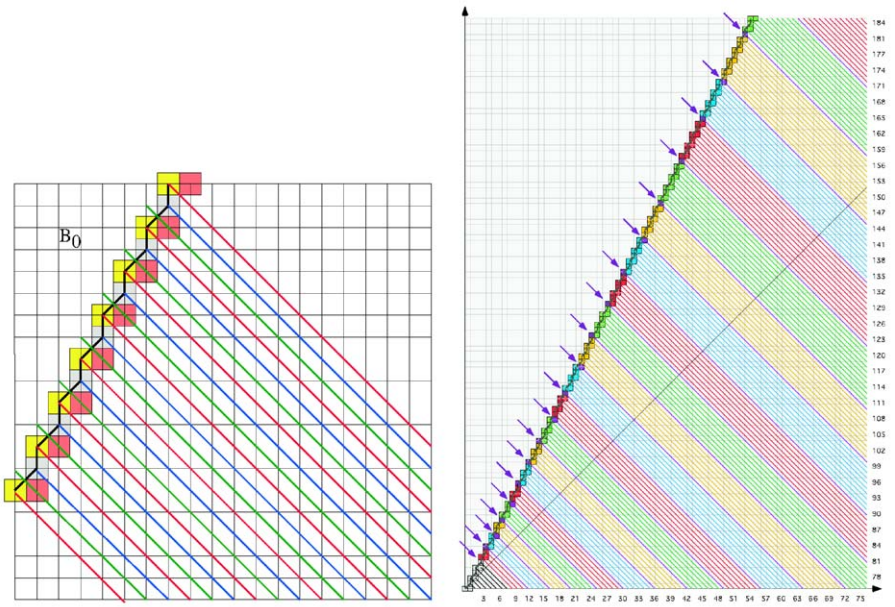


Fig. 8. Setting up signal B_0 (finished): (a) setting up signal B_0 (by means of states); (b) spires finally marked, constituting B_0.

It is easy to observe that there are three possibilities of meeting with S_2 for a signal carrying the information "beginning of a spire" as shows Fig. 6. Each of these cases gives birth to three ways to generate a new segment of length k and also three ways to generate a new segment of length $(k+1)$. That implies to study 3^3 cases, what would be very tedious to exhaustively present here. Fig. 7 illustrates one case, while Fig. 8(a) shows how signal B_0 is build by means of states and Fig. 8(b) gives a global representation of it.

In order to show how to simulate the first step of computation of $\mathscr{A}_\chi$ on $\mathscr{B}$, and in particular to analyze how to translate through the behavior of $\mathscr{B}$ the interactions between consecutive spires in the behavior of $\mathscr{A}_\chi$, we need some notations.

(c) *Notations*: The set of states of $\mathscr{B}$ will be $Q_S \times Q_{\nwarrow}^5 \times Q_{\uparrow}^{11} \times Q_{\nearrow}^2 \times Q_{\rightarrow}^2 \times Q_{\leftarrow}^2 \times Q_{\|}$ where elements of:

- Q_S set up moves of signals in $S_{k,0}$, thus they include elements indicating the four different "sides" of the spires,
- $Q_{\nwarrow}$, $Q_{\nearrow}$ and $Q_{\uparrow}$ represent letters of the alphabet of the language L, when they move to the left at maximal speed, to the right (at a speed which will depend on signal B_t) or stay in place.

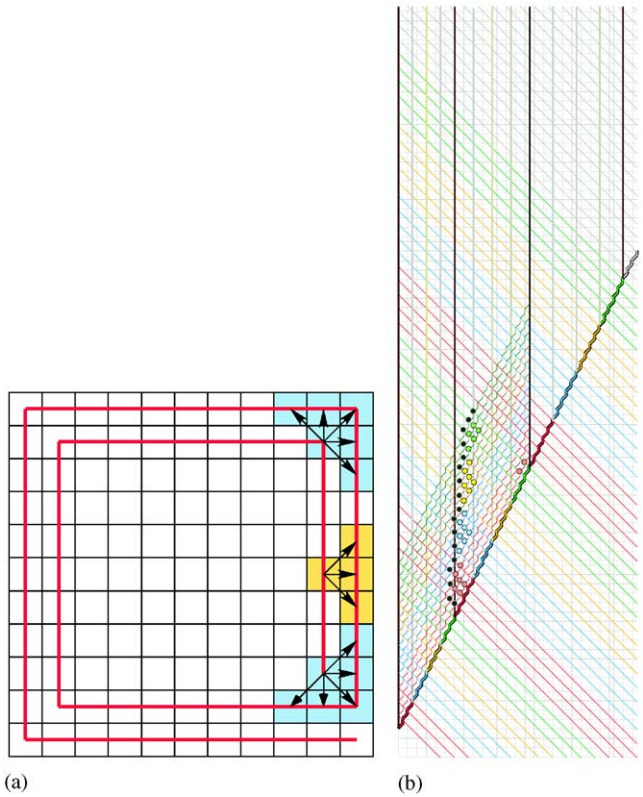


Fig. 9. Interactions between the $(k-1)$th spire and the kth one in the proof of Theorem 7, point (d)(1): (a) area under scope of the $(k-1)$th spire; (b) interactions between data from the $(k-1)$th spire via $Q_{\nearrow}$ and letters from the kth spire via $Q_{\nwarrow}$.

- $Q_{\rightarrow}$ mark sites where data of the $(k-1)$th spire interact with data of the kth one.
- $Q_{\leftarrow}$ mark sites where data of the $(k+1)$th spire interact with data of the kth one.

(d) *Interactions between the* $(k-1)$*th and* k*th spires*: The nine first cells are subject of a special processing and cell 0 acts as the nine first cells of the thread, cell 8 acts as the last cell of the first spire.

- Fig. 9 shows on (a) how cells on the $(k-1)$th spire influence cells on the kth one. Outside the "corner cells" and the first cell of the spire, which, respectively, influence five cells of spire k, cells of spire $(k-1)$ influence three cells of spire k. Data on B_0 both stay on the same site (states in $Q_{\uparrow}$) and are sent at maximal speed to the left (states in $Q_{\nwarrow}$). When the latter meet some signal V_k, information goes on to the left but is also sent by another signal, at speed 2, to the right (states in $Q_{\nearrow}$).

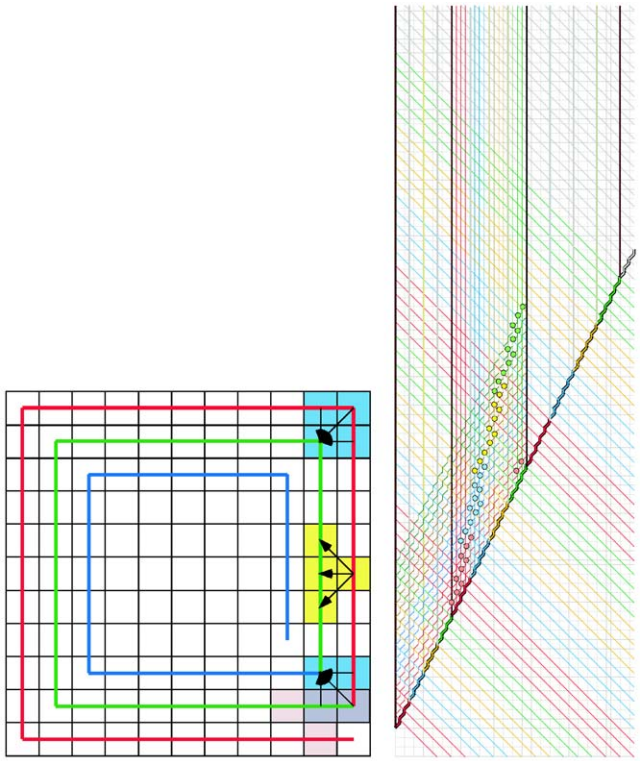


Fig. 10. Interactions between the $(k-1)$th spire and the kth one in the proof of Theorem 7, point (d)(2): (a) area under scope of the $(k+1)$th spire; (b) interactions between data from the $(k-1)$th spire via $Q_{\nearrow}$ and the kth one via $Q_{\uparrow}$.

Fig. 9(b) shows where letters[6] of the $(k-1)$th spire, moving to the right and encoded in $Q_{\nearrow}$ meet letters coming from the kth spire and encoded in $Q_{\nwarrow}$. These meeting sites are encoded in $Q_{\rightarrow}$. Particular sites corresponding to the "beginning and the end" of a spire are special (dark points on the figure): they define the signal Σ_k which disappears as soon as a signal from $Q_{\nwarrow}$ arrives what means that letters lay on the next spire.

- Fig. 10(a) shows how cells on the $(k+1)$th spire influence cells on the kth one at strategic points.

  Fig. 10(b) shows where letters of the $(k-1)$ spire, coming back to the right (states in $Q_{\nearrow}$) meet letters, encoded in $Q_{\uparrow}$, of the kth spire. Signals encoded in $Q_{\uparrow}$ and $Q_{\nearrow}$ carry on their information until it becomes useless.

(e) *Interactions between the kth and* $(k+1)$*th spires*: The situation is quite similar to the last point. Fig. 11(a) shows how the $(k+1)$th spire influences the kth one at special places.

[6] Actually, it is shortening of "information characterizing initial letters".

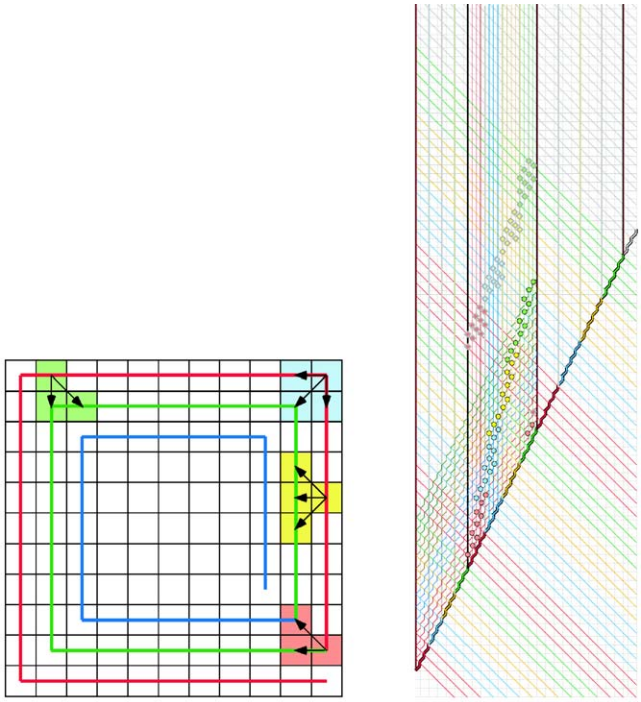


Fig. 11. Interactions between states of the kth spire and those of the $(k+1)$th one in the proof of Theorem 7, point (f): (a) area under scope of the $(k+1)$th spire; (b) interactions between data from the kth spire via $Q_{\uparrow}$ and data from the $(k+1)$th one via $Q_{\nwarrow}$.

Fig. 11(b) shows where information coming from spires $k-1$ and k via states of $Q_{\uparrow}$ meets data coming from spire $(k+1)$ via states of $Q_{\nwarrow}$. These meeting sites are encoded in $Q_{\leftarrow}$.

(f) *Simulation of the first transition of* $\mathscr{A}_{\chi}$: At the end of the simulation of the first computation step of $\mathscr{A}_{\chi}$, one gets a space-time diagram of $\mathscr{B}$ the main features of which are:

- The nine first cells have initialized the process.
  Via states of $Q_{\|}$, the space-time diagram of $\mathscr{B}$ is cut into zones. When the mark "to be the first cell of a spire" appears on B_0, two signals are sent: one (A_k of point (a)) at maximal speed to the left by means of state $q_{\|,\nwarrow}$, the other one staying on the site (V_k of point (a)) by means of state $q_{\|,\uparrow}$. The mentioned zones are the area determined by two consecutive convenient A_k cut by two consecutive convenient V_k (see point (a)).
- If one focuses on the area determined by the space corresponding to a spire k,
  – the states of $Q_{\leftarrow}$, taking into account letters of the $(k+1)$th spire (in gray bordered black in Fig. 12) constitute the significant data to be used for the simulation of

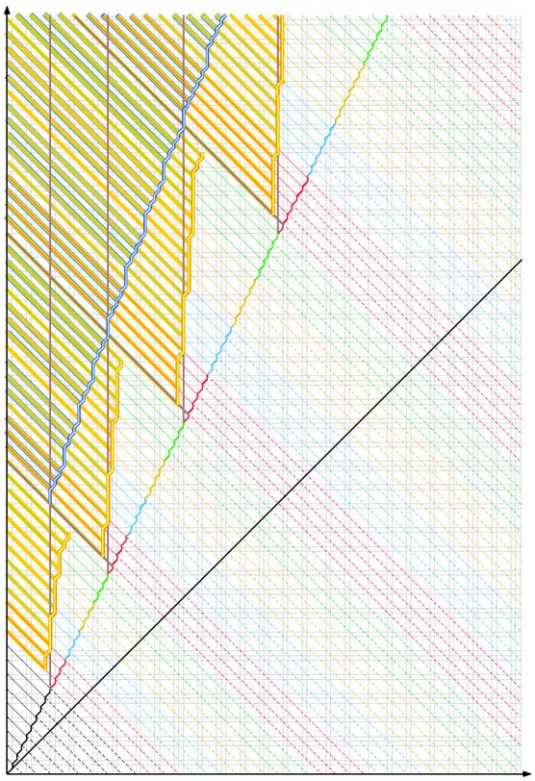


Fig. 12. Simulation of the first transition of $\mathscr{B}$ in the proof of Theorem 7, point (f).

the next transition of $\mathscr{A}_\chi$. The signal B_1 they determine will play the role of B_0, and they will be sent to the left via states in $Q_{\nwarrow}$ (see point (e)),

– from sites in states of $Q_{\rightarrow}$, on signals $\Sigma_{0,k}$, which anticipate the end of the input word (in yellow bordered black in Fig. 12), they will be sent to the left via $Q_{\leftarrow}$ (see point (d)) in order to carry information that the corresponding spire is the one with the end of the input word.

(g) *Simulation of transitions of* $\mathscr{A}_\chi$: In order to simulate the second computation step of $\mathscr{A}_\chi$, and then to iterate the process, signal B_1 will play the role of B_0. The process is quite identical as the one described above except for the moves of states in $Q_{\nearrow}$: they will go on moving to the right but now parallel to B_1. As in the general case we do not know moves of B_t, new signals ς are used, which indicate that $S_{k,t}$ has gone to the right (these signals run to the left and are killed by next signal $S_{k,t+1}$) (see Fig. 13). Fig. 14 shows the simulation on $\mathscr{B}$ of the two first computation steps of $\mathscr{A}_\chi$. □

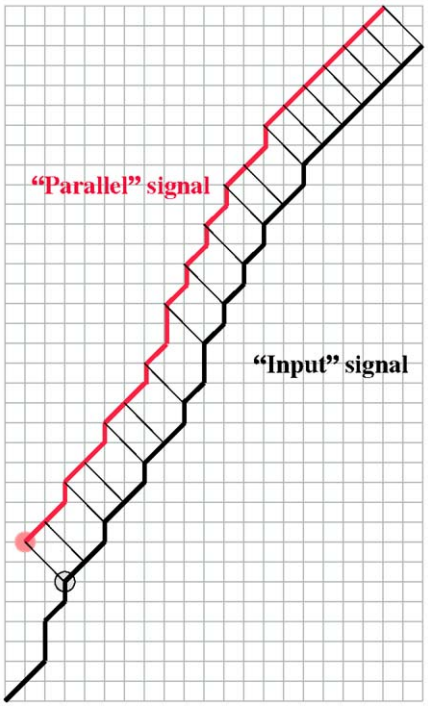


Fig. 13. Construction of a "parallel" signal in the proof of Theorem 7, point (g).

5. Conclusion

We have proved that $\mathscr{R}_\chi \subsetneq \mathscr{R}_{1D}$.[7] That is that two-dimensional cellular automata with the Archimedean thread are faster than one-dimensional cellular automata for some languages but not for all languages in $\mathscr{R}_{1D}$. How important is the gap?

The Archimedean thread certainly is a very special one, nevertheless this result confirms that the way input data are space-distributed is not immaterial, and the theorem allows to grasp what can be the difficulty to understand and master the multiplicity of types of parallelism. It could be interesting to study other threads—among them non-recursive ones—and to compare them via the real-time classes they determine.

Besides, studying classes $\mathscr{R}_\tau$ for different threads τ as well as their comparisons with the class $\mathscr{R}_{1D}$ could give finer knowledge on the way parallelism acts. Moreover, one can wonder whether it would be possible to exhibit some significant hierarchy $(\mathscr{R}_{\tau_i})_i$ inside $\mathscr{R}_{1D}$?

[7] Let us remark that our proof implies that $\mathscr{R}'_\chi \subsetneq \mathscr{R}_{1D}$ where $\mathscr{R}'_\chi$ denotes the languages real-time recognized by two-dimensional cellular automata with the Von Neumann neighborhood.

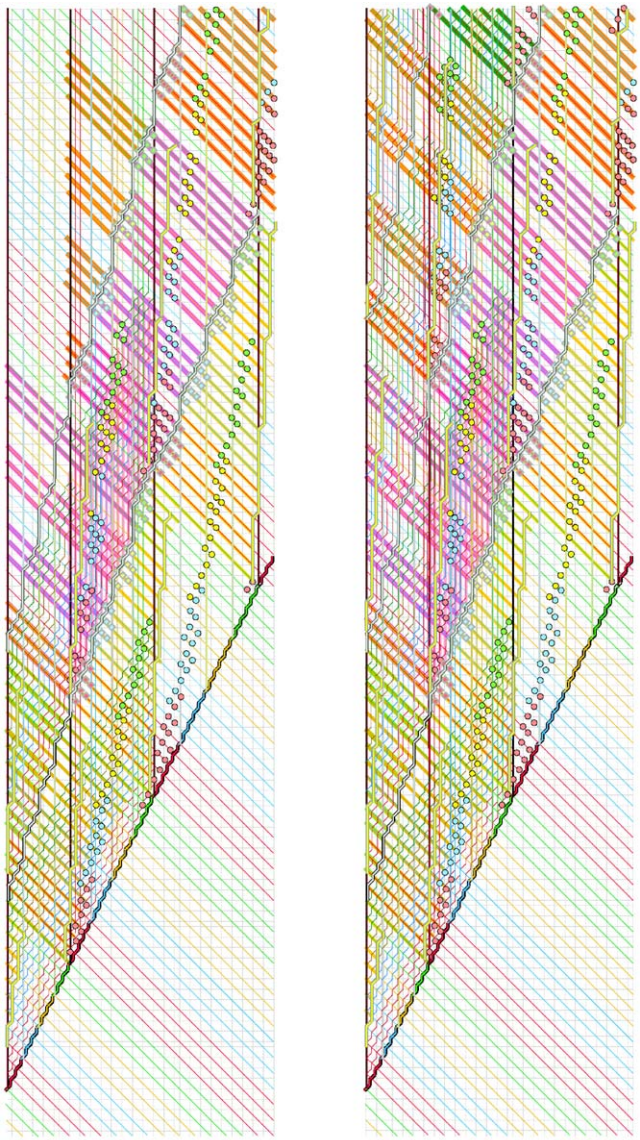


Fig. 14. Simulation of $\mathscr{B}$ in the proof of Theorem 7, point (g). Moves of states of $Q_{\nearrow}$ are only shown for the fourth spire: (a) simulation of the two first transitions of $\mathscr{A}$ on the fourth spire; (b) simulation of transitions of $\mathscr{A}$ by $\mathscr{B}$.

Does it make sense to look for one thread τ such that $\mathscr{R}_\tau = \mathscr{R}_{1D}$? Assuming existence of such a thread, for $\mathscr{R}_\tau(n) = O(n)$ to hold would show that using τ to take advantage of the space is inefficient, but $\mathscr{R}_\tau(n) = o(n)$ would mean that τ expresses a sort of universality and show that $\mathscr{R}_{1D}$ is a "true" complexity class.

References

[1] W. Bucher, K. Culik, On real time and linear time cellular automata, RAIRO 18 (4) (1984) 307–325.
[2] S. Cole, Real time computation by n dimensional iterative arrays of finite-state machines, IEEE Trans. Comput. 18 (4) (1969) 349–362.
[3] M. Delorme, J. Mazoyer, Languages recognition on cellular automata, in: J. Almeida (Ed.), Semigroups, Automata and Languages, World Scientific, Singapore, 1994, pp. 85–100.
[4] M. Delorme, J. Mazoyer, Reconnaissance parallèle des langages rationels sur automates cellulaires plans, Theoret. Comput. Sci. 281 (2002) 251–289.
[5] O. Ibarra, Computational complexity of cellular automata: an overview, in: M. Delorme, J. Mazoyer (Eds.), Cellular Automata, a Parallel Model, Kluwer Academic, Dordrecht, 1994, pp. 85–100.
[6] J. Mazoyer, V. Terrier, Signals in one dimensional finite automata, Theoret. Comput. Sci. 217 (1999) 53–80.
[7] A. Smith, Real time languages by one-dimensional cellular automata, J. Comput. System Sci. 6 (1972) 233–253.
[8] V. Terrier, Two-dimensional cellular automata recognizer 218 (1999) 325–346.